Bloque III: El nivel de transporte

Tema 6: Conexiones TCP

Índice

- Bloque III: El nivel de transporte
  - Tema 6: Conexiones TCP
    - Establecimiento de conexión
    - Finalización de conexión
    - Establecimiento y finalización simultáneos
    - Diagrama de estados
    - Segmentos de reset

- **Lecturas recomendadas:**
  - Capítulo 3, secciones 3.5.1 y 3.5.6 de “Redes de Computadores: Un enfoque descendente”. James F. Kurose, Keith W. Ross. Addison Wesley.
  - Capítulo 13 de “TCP/IP Illustrated, Volume 1: The Protocols”, W. Richard Stevens, Addison Wesley.

Conexión TCP: Establecimiento

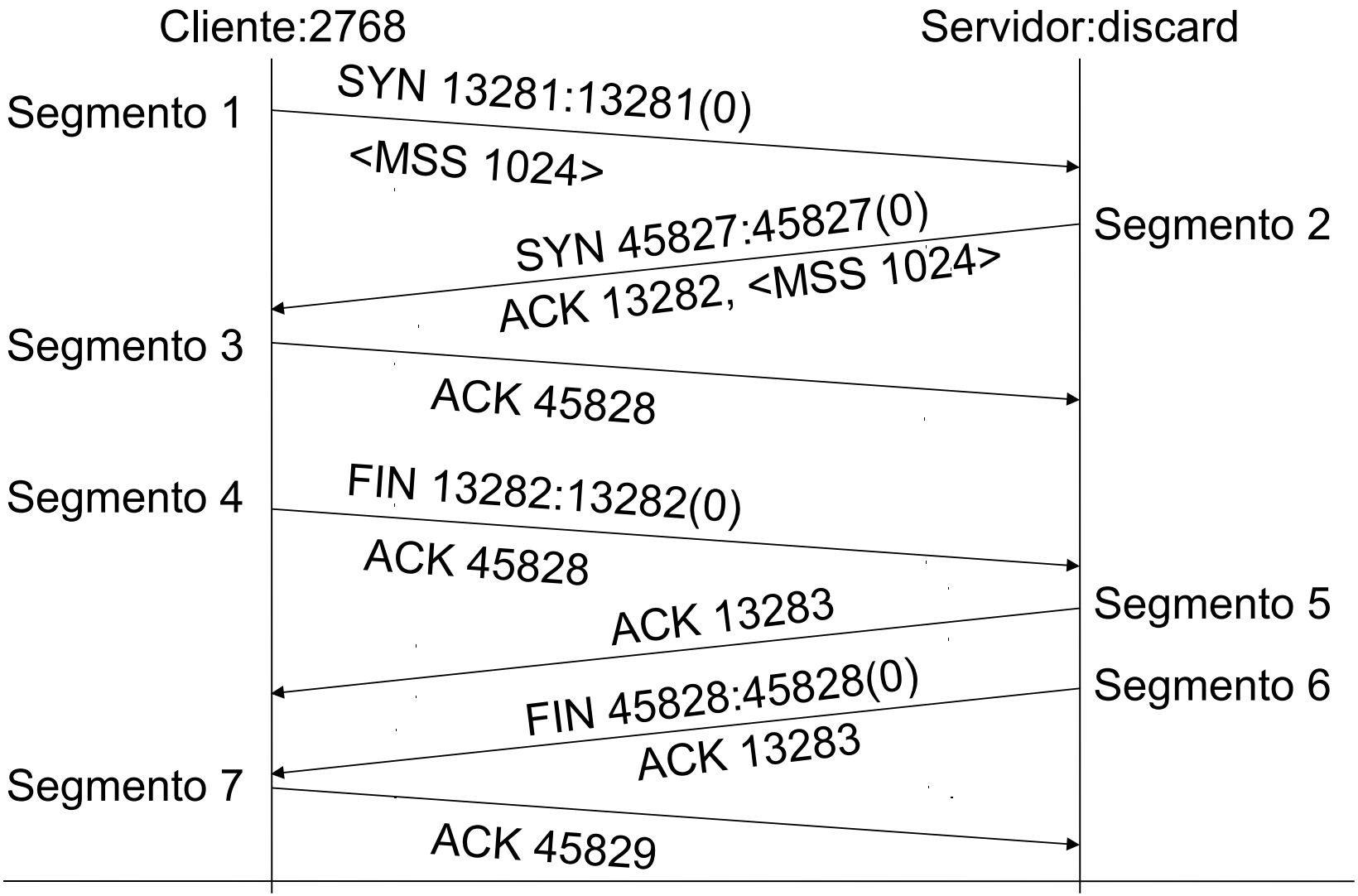
- Las conexiones las inicia, normalmente, el cliente (**apertura activa**) → El servidor hace una **apertura pasiva**.
- Protocolo de establecimiento de conexión (**Three-Way Handshake**):
  - El emisor (cliente) envía un segmento SYN indicando el nº de puerto del servidor al que quiere conectarse y el nº de secuencia inicial (segmento 1).
  - El servidor responde con su propio segmento SYN que contiene el nº de secuencia inicial del servidor (segmento 2). También confirma (ACK) el SYN del cliente + 1 (los mensajes SYN consumen un nº de secuencia).
  - El cliente confirma el SYN del servidor con un nº de ack igual al ISN del servidor +1 (segmento 3).
- **Número de secuencia inicial** (**ISN**):
  - Cada extremo selecciona su ISN al establecerse la conexión.
  - Se obtiene (pseudo)aleatoriamente.
  - Objetivo: evitar que segmentos “antiguos” (de otra conexión igual) se confundan con los actuales → Reencarnación.

Conexión TCP: Finalización

- Se intercambian 4 segmentos para cerrar una conexión.
  - Una conexión TCP es full-duplex y cada dirección se cierra independientemente.
  - Cada extremo envía un FIN cuando ha finalizado el envío de datos → El otro extremo puede continuar enviando datos (half-close).
- El extremo que envía el primer FIN realiza el cierre activo, y el otro extremo el cierre pasivo.
  - Cualquiera de los dos extremos puede iniciar el cierre.
- Protocolo de finalización de conexión:
  - El cliente finaliza la aplicación → El cliente TCP envía un FIN (segmento 4) con el número de secuencia correspondiente (cierre del flujo de datos cliente a servidor).
  - El servidor responde con un ACK (segmento 5) del nº de secuencia + 1 (los mensajes FIN consumen un nº de secuencia).
  - A continuación, el servidor envía un FIN (segmento 6).
  - El cliente confirma la recepción del FIN, con un ACK del nº de secuencia recibido + 1 (segmento 7).

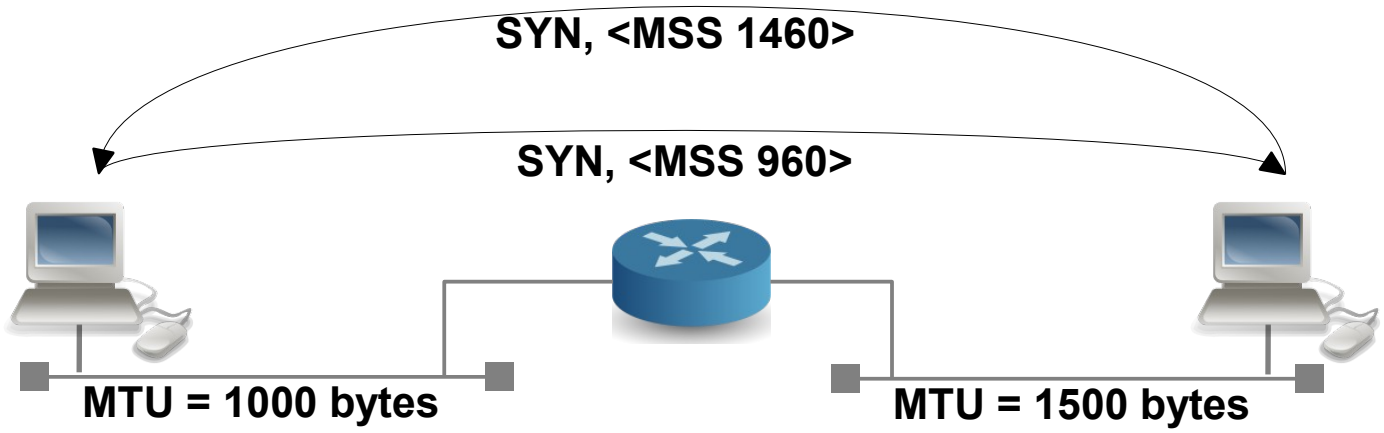
¿Qué segmento puedo eliminar de la finalización sin que afecte?

Conexión TCP



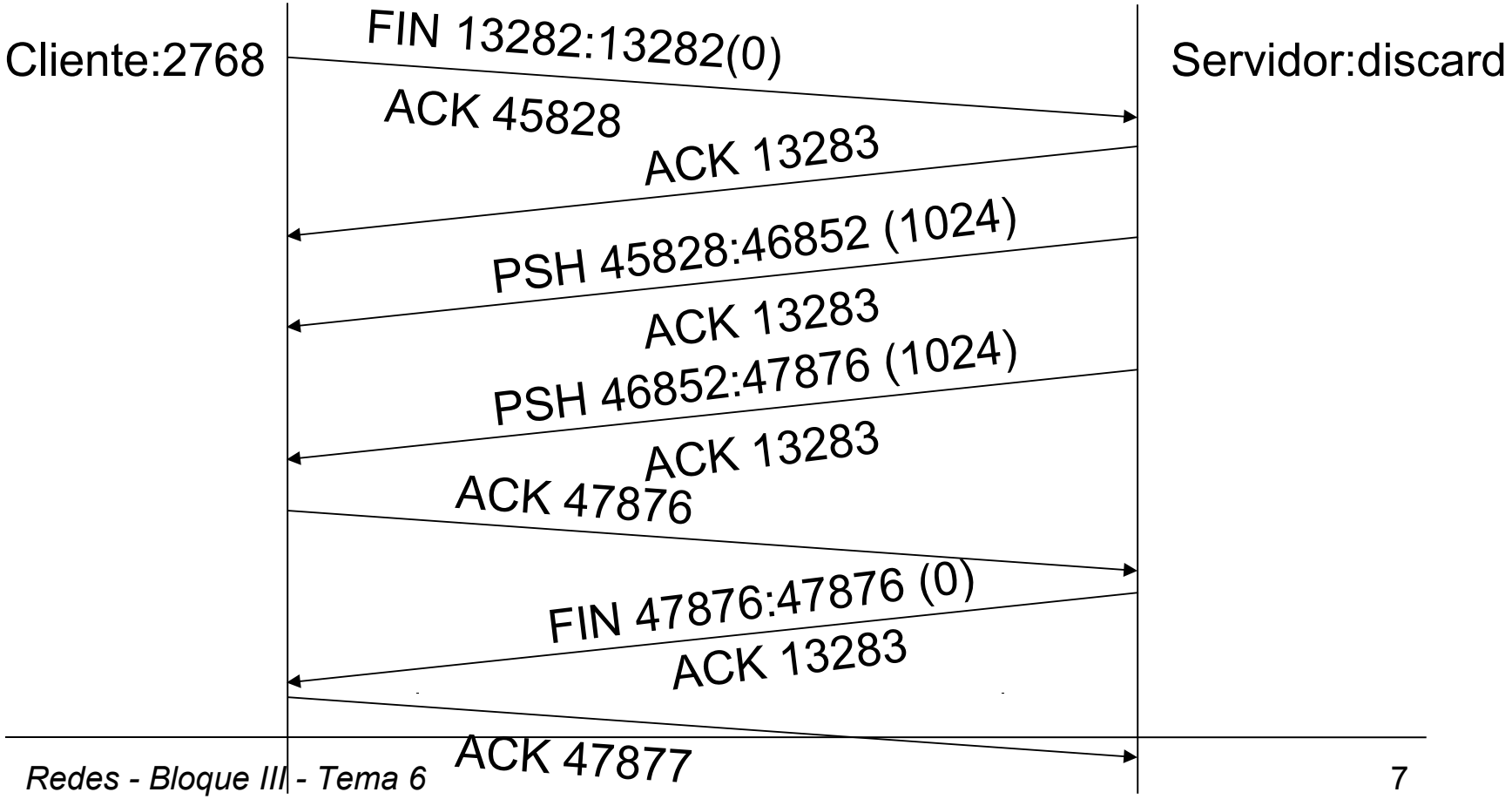
Conexión TCP: MSS

- **Maximum Transmission Unit (MTU)**: número máximo de bytes de datos que puede enviar el nivel de enlace.
- **Maximum Segment Size (MSS)**: indica el número máximo de bytes de datos que le conviene recibir a cada extremo (para evitar la fragmentación IP).
- Cuando se establece una conexión TCP, cada extremo anuncia el MSS que espera recibir:
  - La opción MSS sólo aparece en un segmento SYN.
  - Si no se declara, se toma por defecto el valor 536 (datagrama IP de 576 bytes).
  - MSS no incluye las longitudes de cabecera IP y TCP (MSS = MTU – 20 – 20).
- En general es preferible un MSS grande que amortice el coste de cabeceras. Pero también interesa evitar fragmentación.
- No se realiza una negociación del MSS, el tamaño de segmento será el menor de los dos.



Conexión TCP: Half-close

- Una parte termina la conexión (da por finalizado su emisión de datos) mientras todavía está recibiendo datos (half-close).
  - No puede enviar más datos, pero sí ACKs, ...
- Pocas aplicaciones se beneficien de esta utilidad.

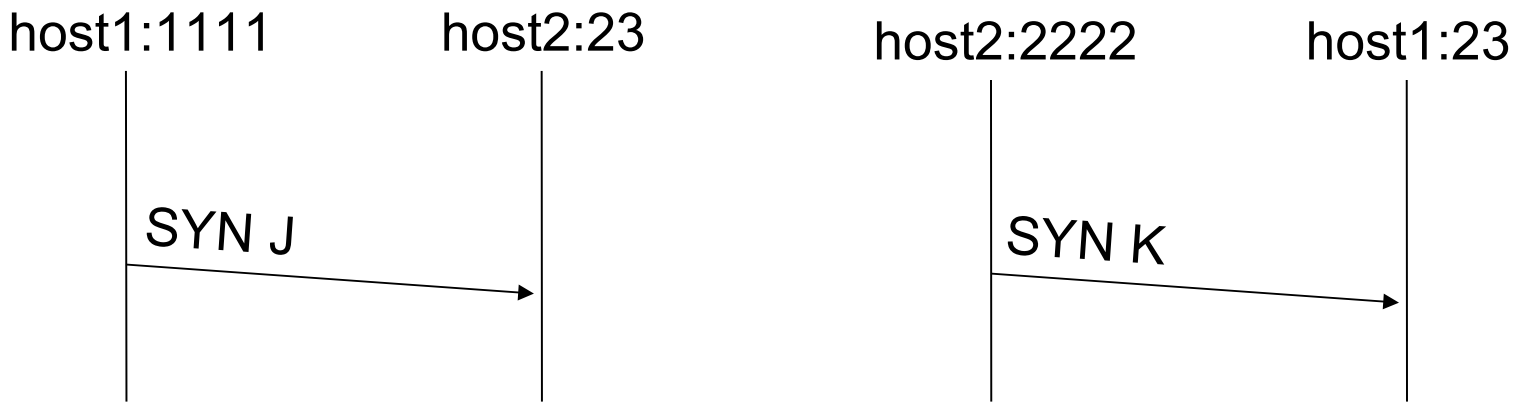


TCP: Establecimiento simultáneo

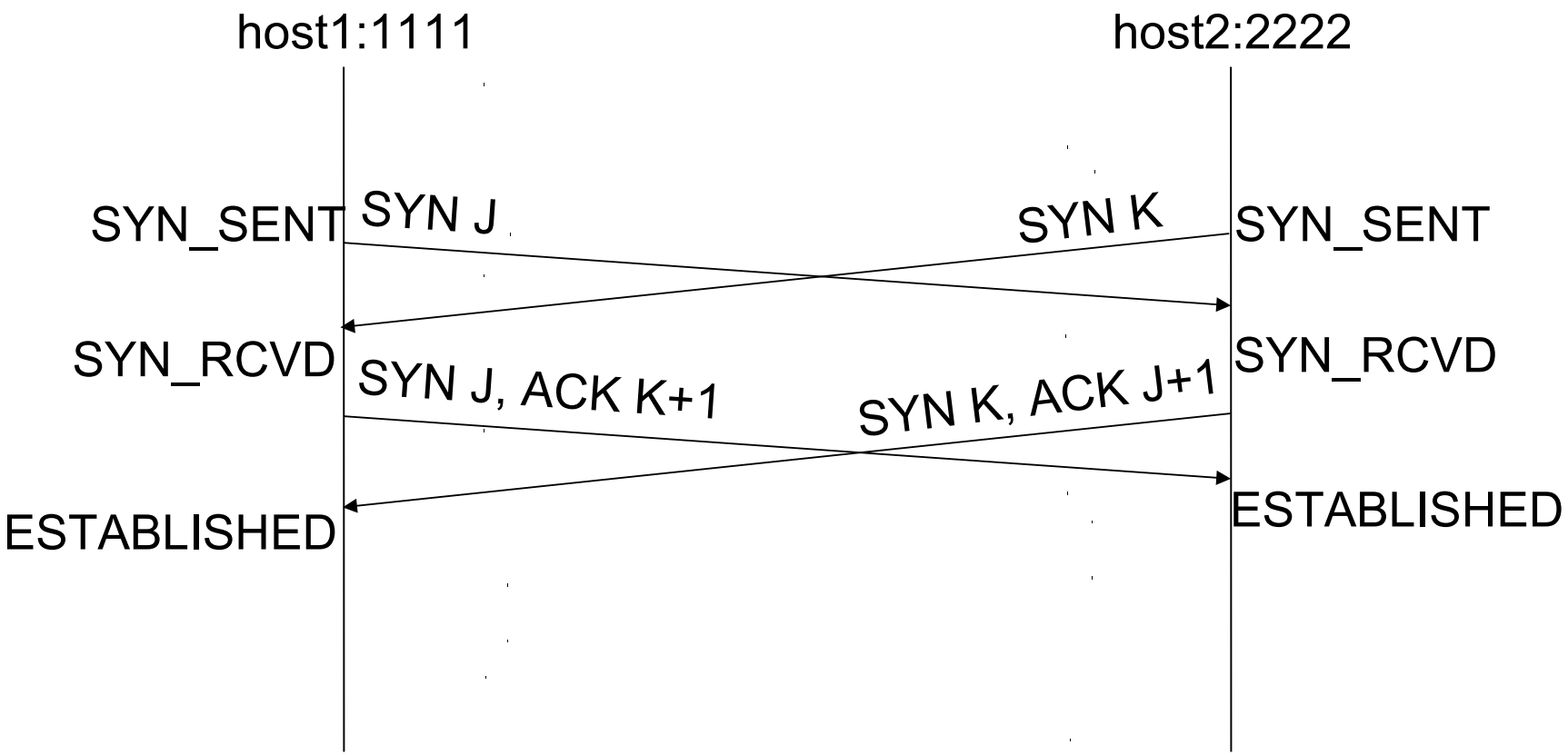
- La posibilidad de establecimiento de conexión simultáneo es mínima aunque posible:
  - Dos aplicaciones en dos hosts se envían mensajes de conexión simultáneamente.
  - No existe una figura clara de cliente y servidor, ahora hay dos "cliente-servidor".
- TCP está diseñado para manejar correctamente esta posibilidad.
- Este tipo de apertura requiere el envío de 4 segmentos (uno más de lo habitual) y sigue algunas variantes sobre el diagrama de transición de estados habitual.

¿Esto es una apertura simultánea?

  - host1% telnet host2 host2% telnet host1

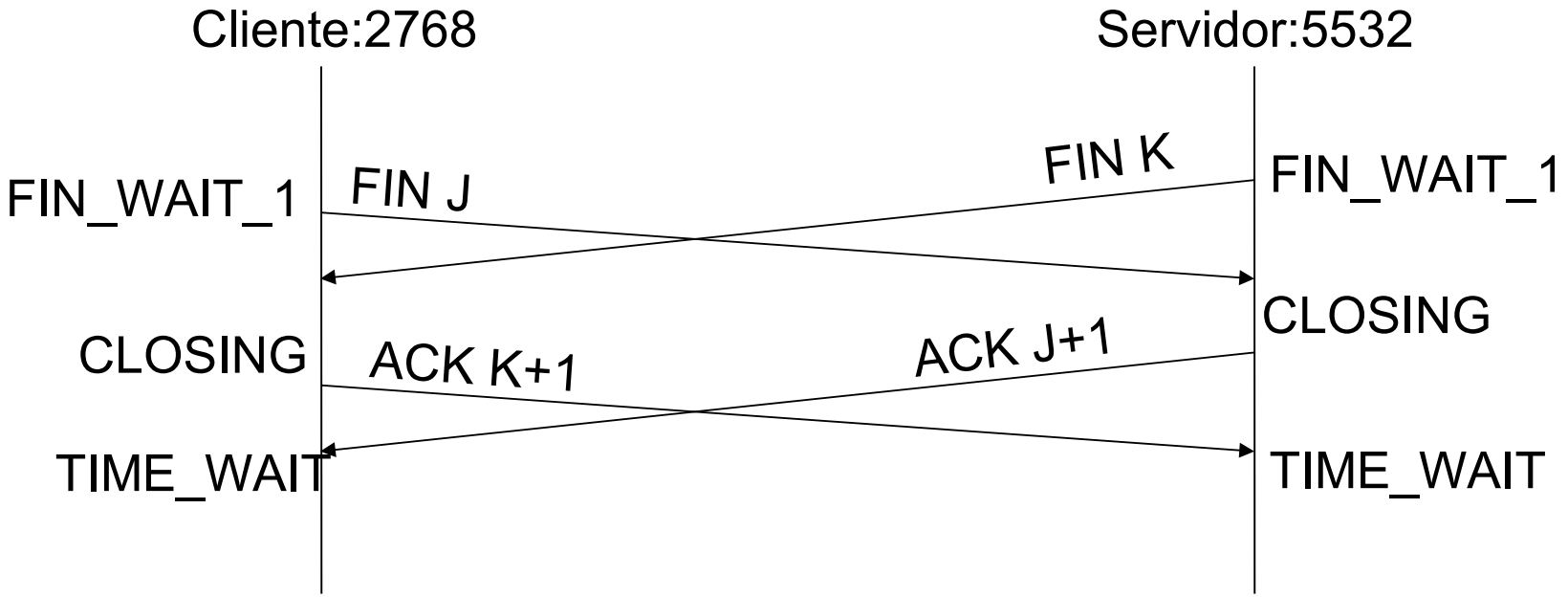


TCP: Establecimiento simultáneo

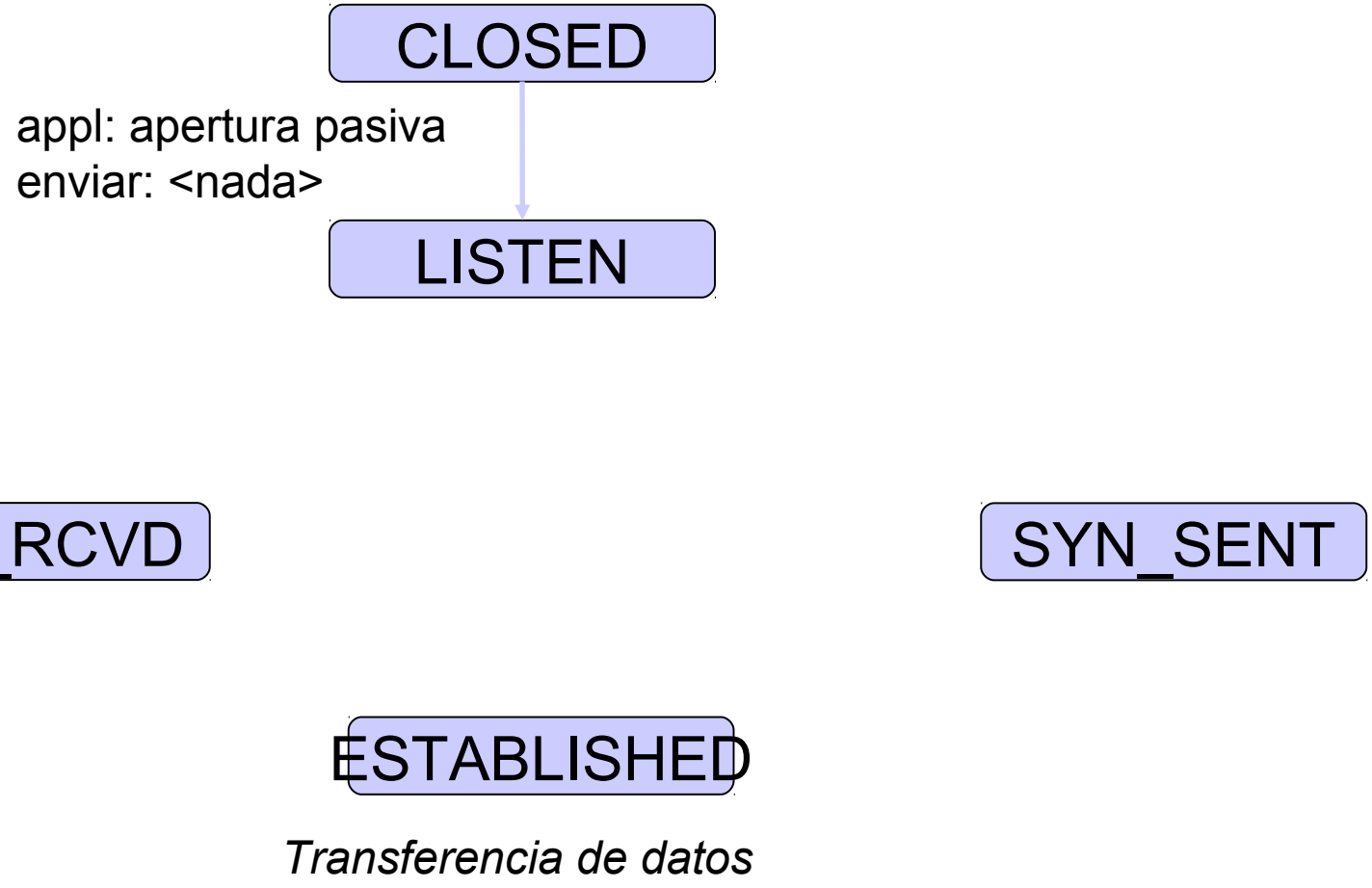


TCP: Cierre simultáneo

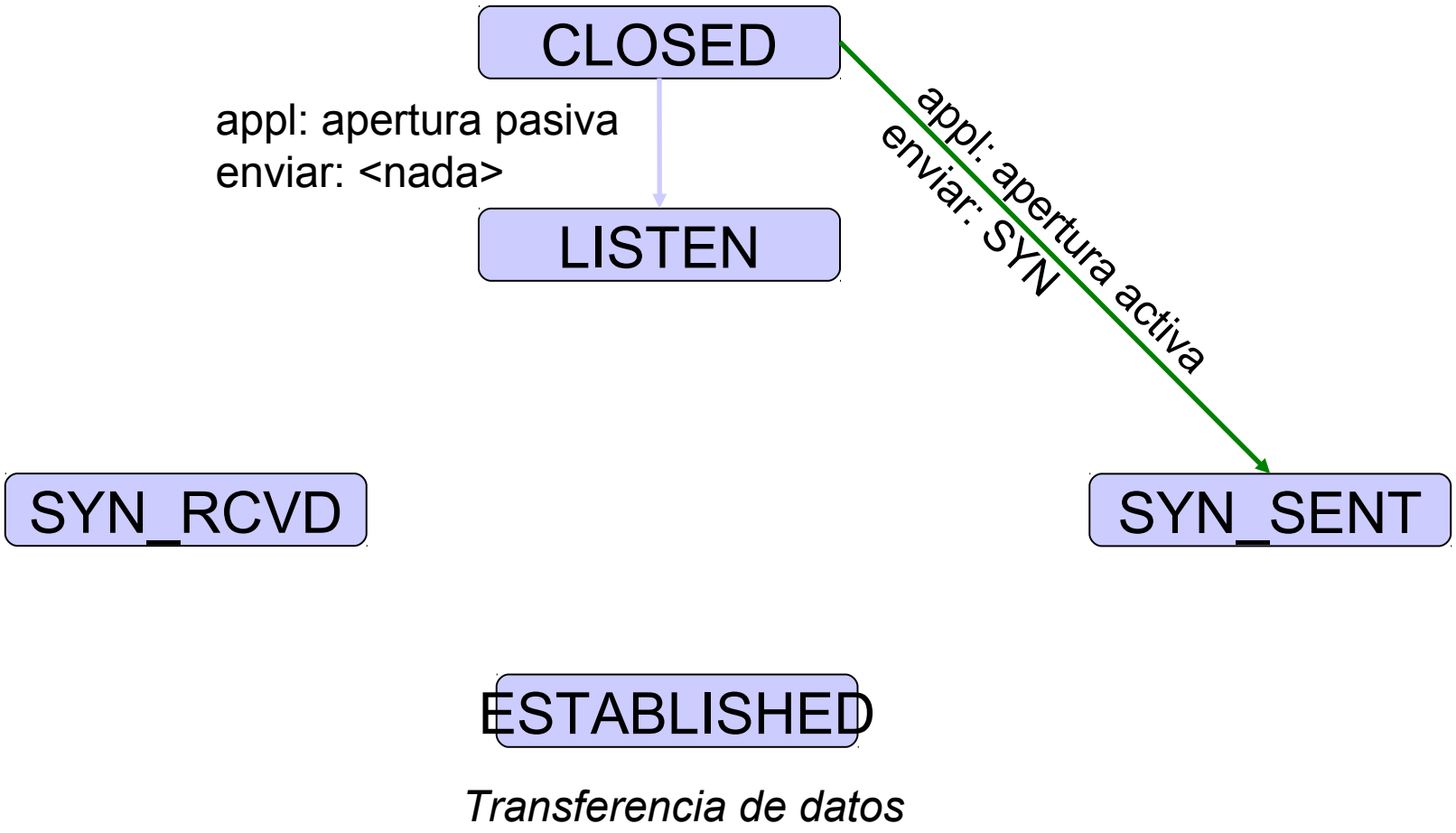
- TCP contempla la posibilidad de dos cierres simultáneos (simultaneous close).
- En este caso el número de segmentos intercambiados es el mismo que en un cierre normal TCP.



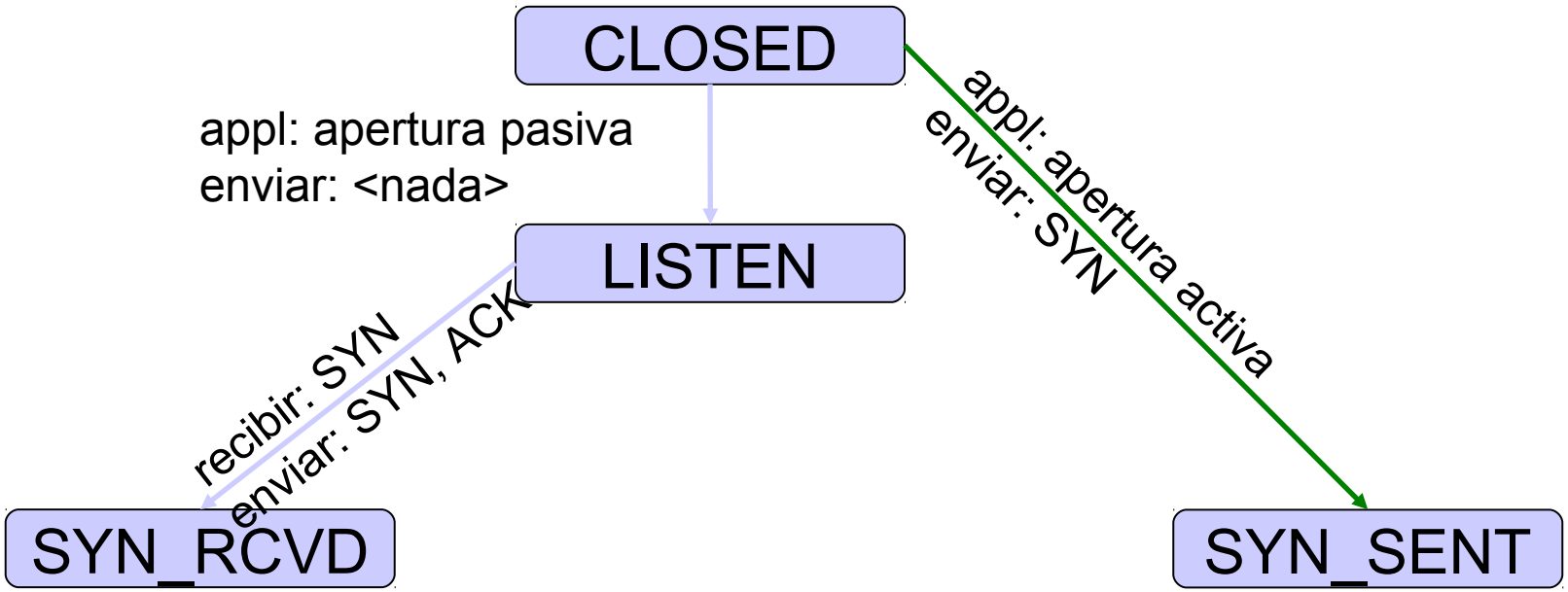
TCP: Diagrama de estados



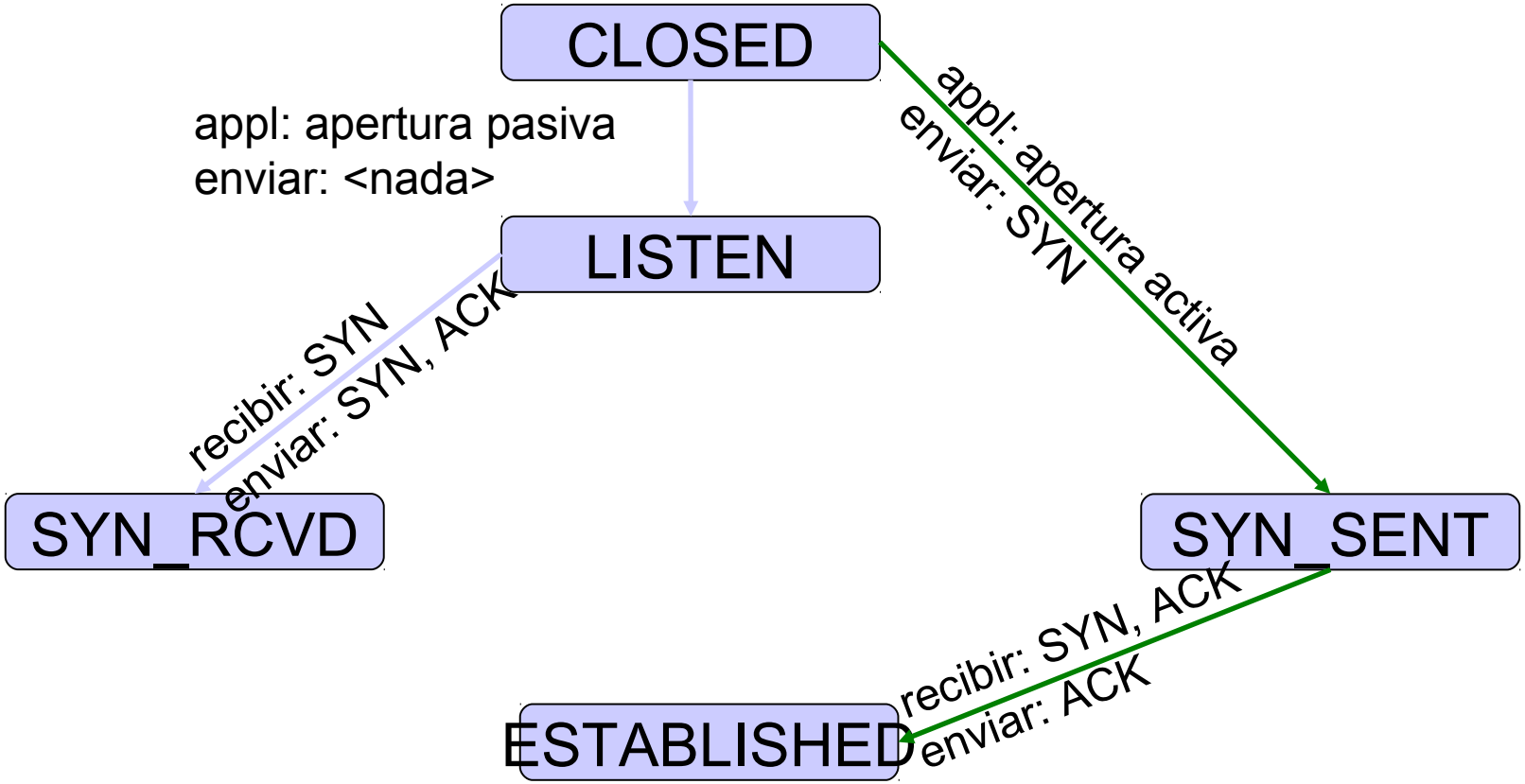
TCP: Diagrama de estados



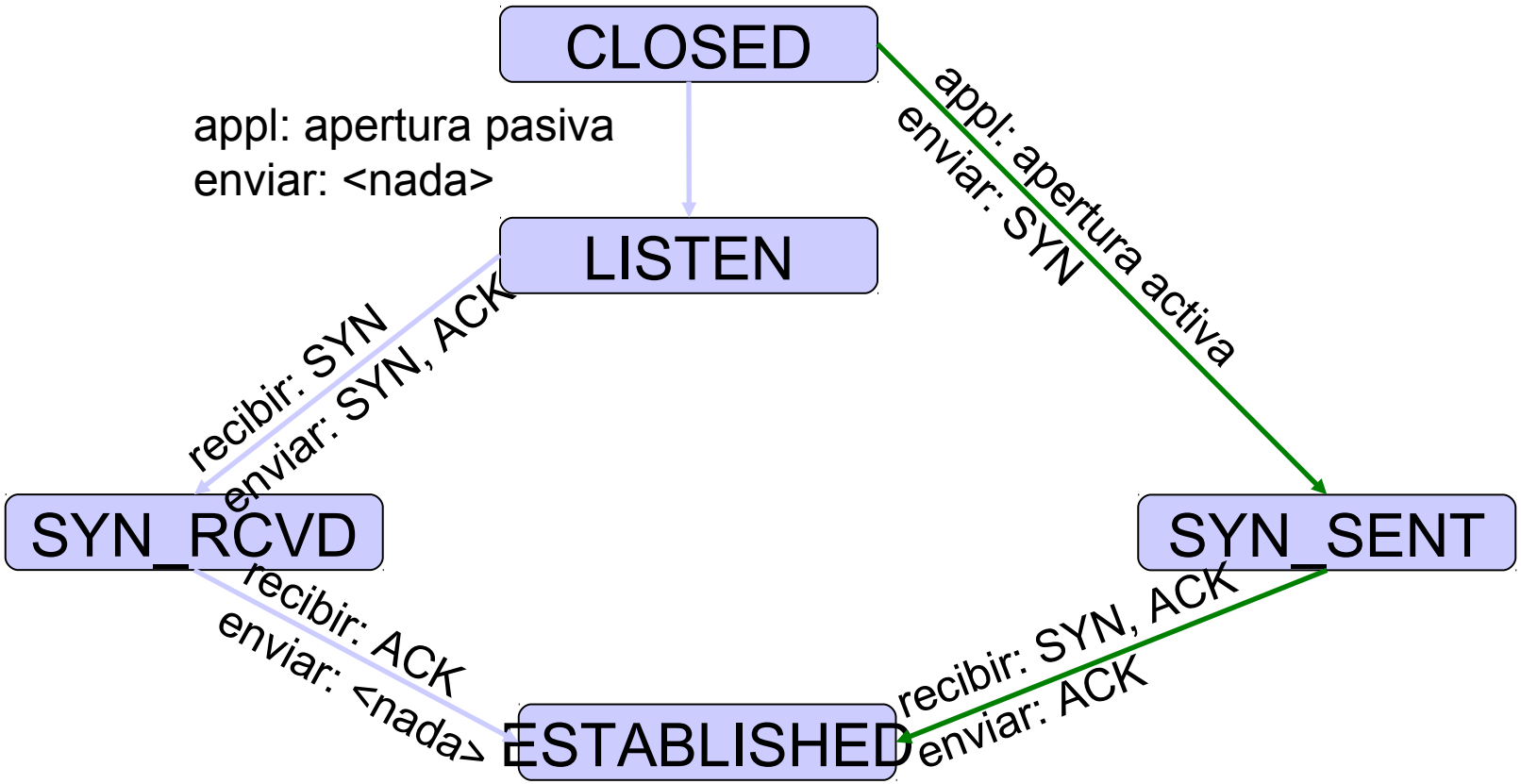
TCP: Diagrama de estados



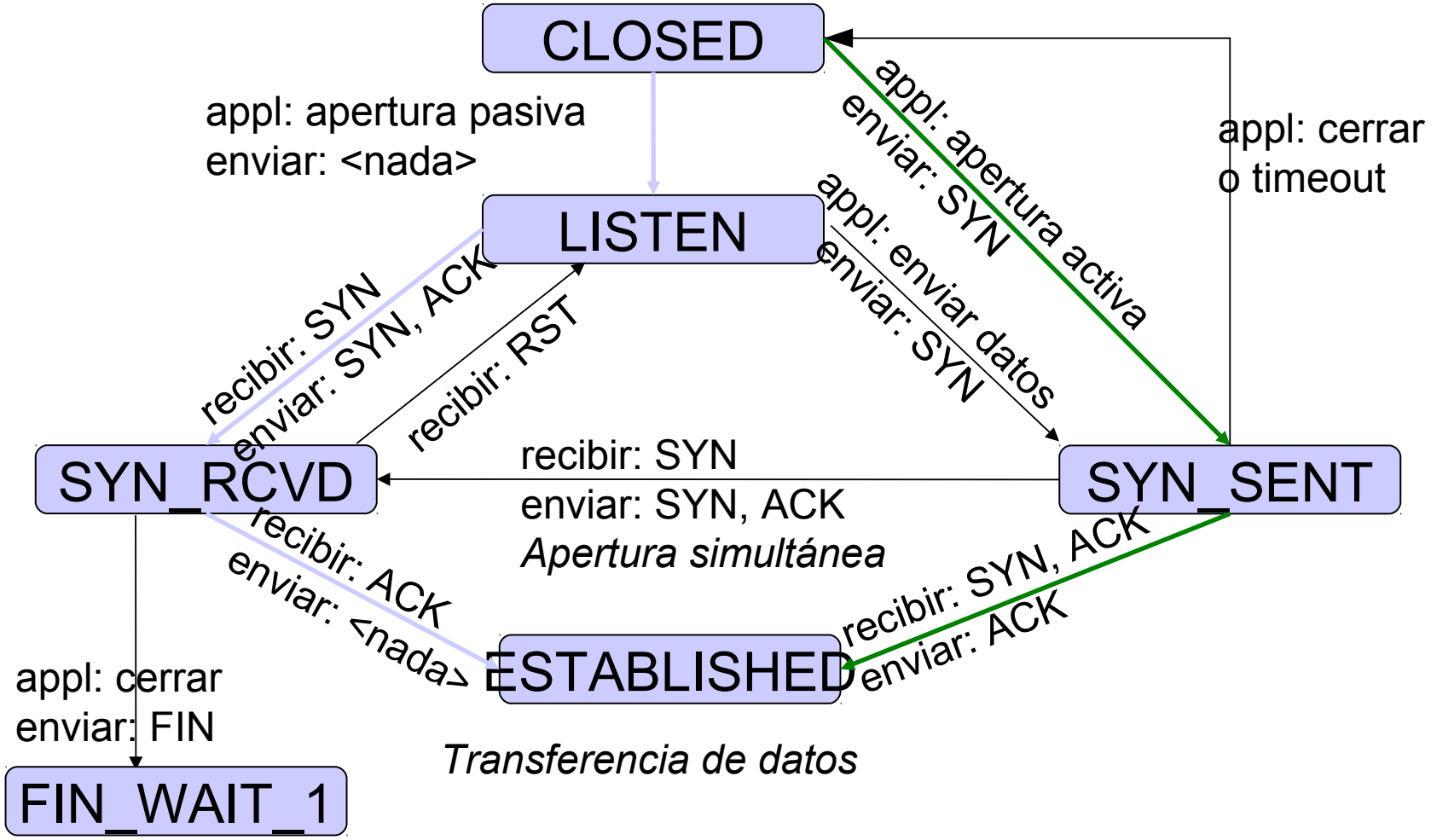
TCP: Diagrama de estados



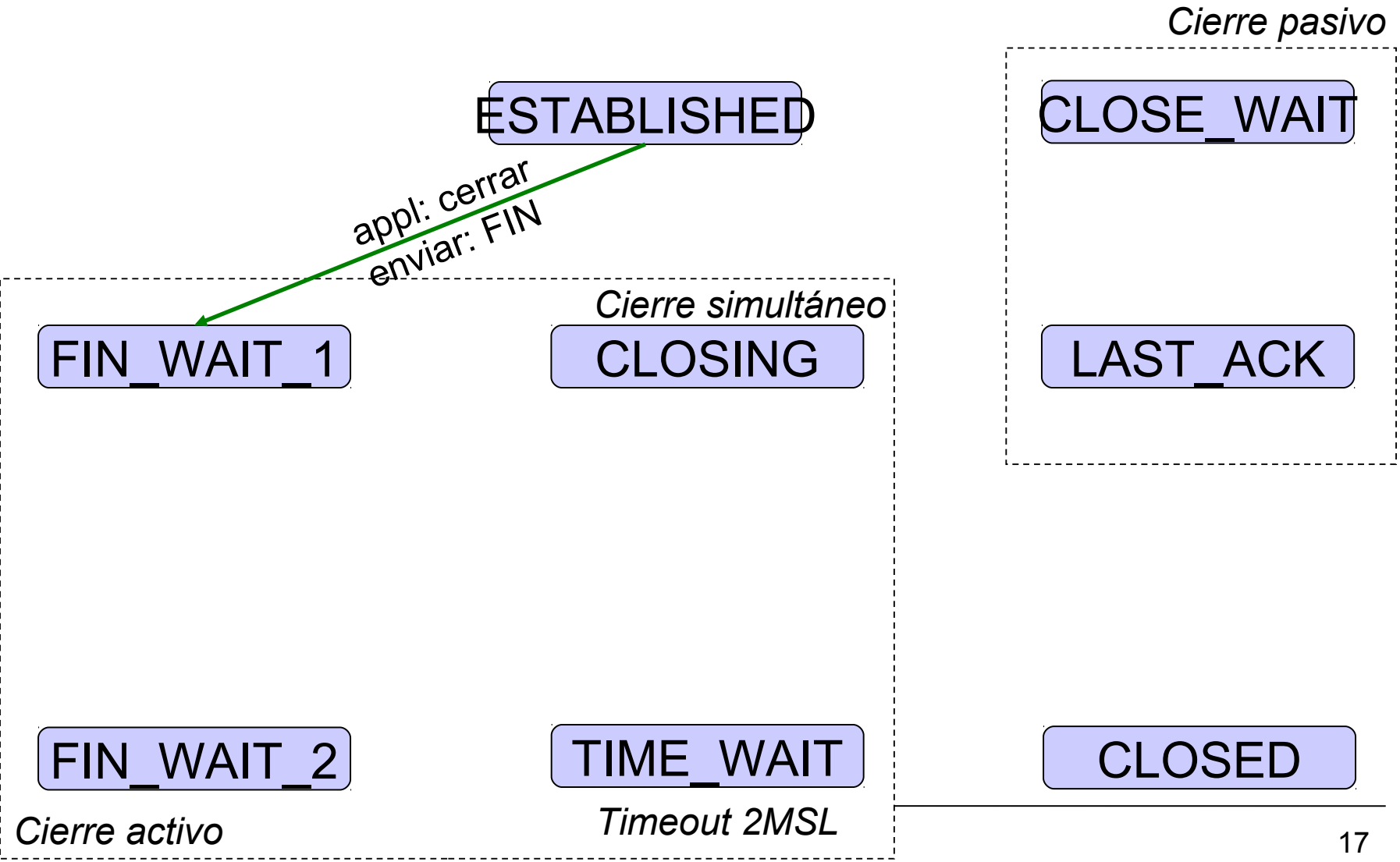
TCP: Diagrama de estados



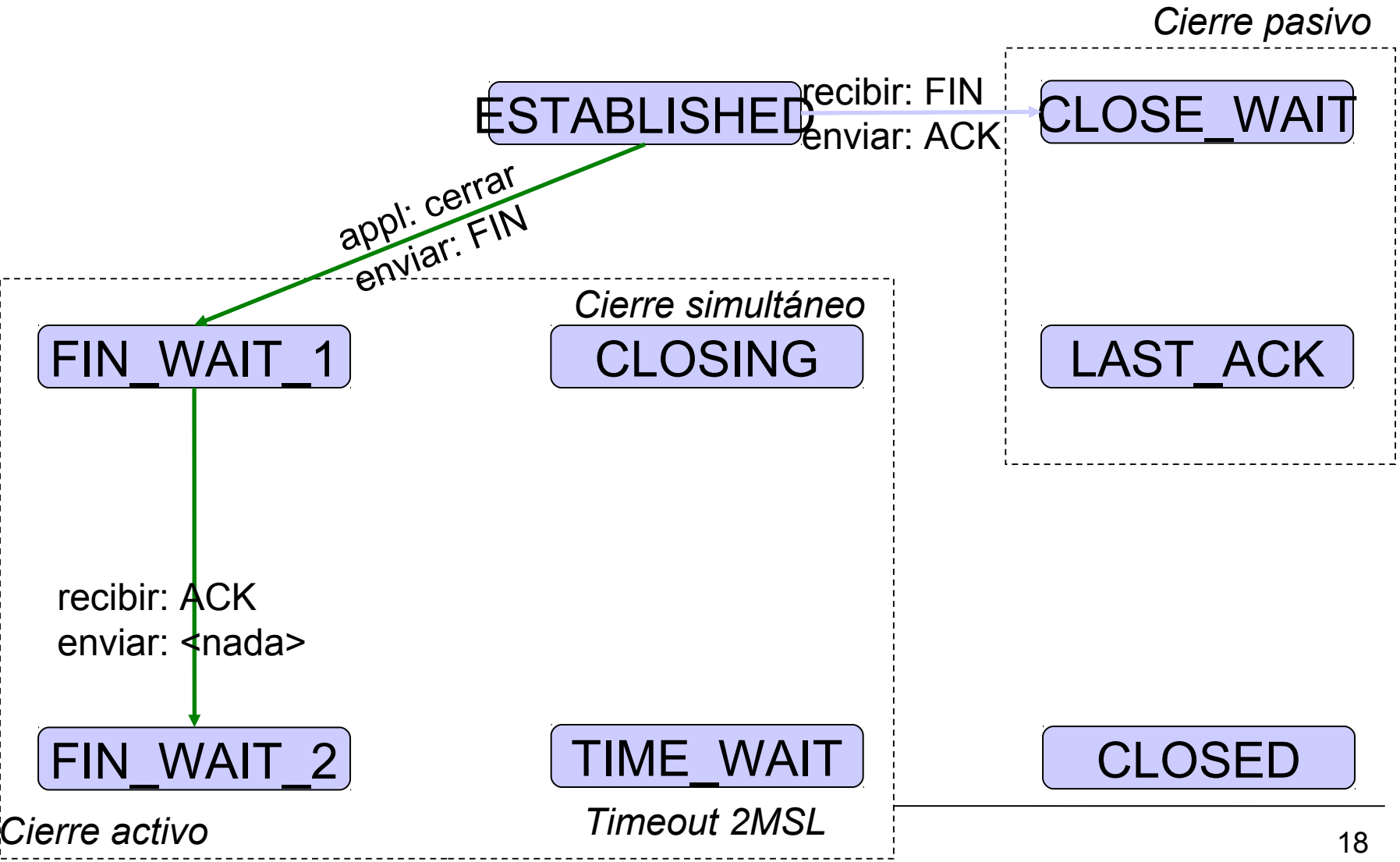
TCP: Diagrama de estados



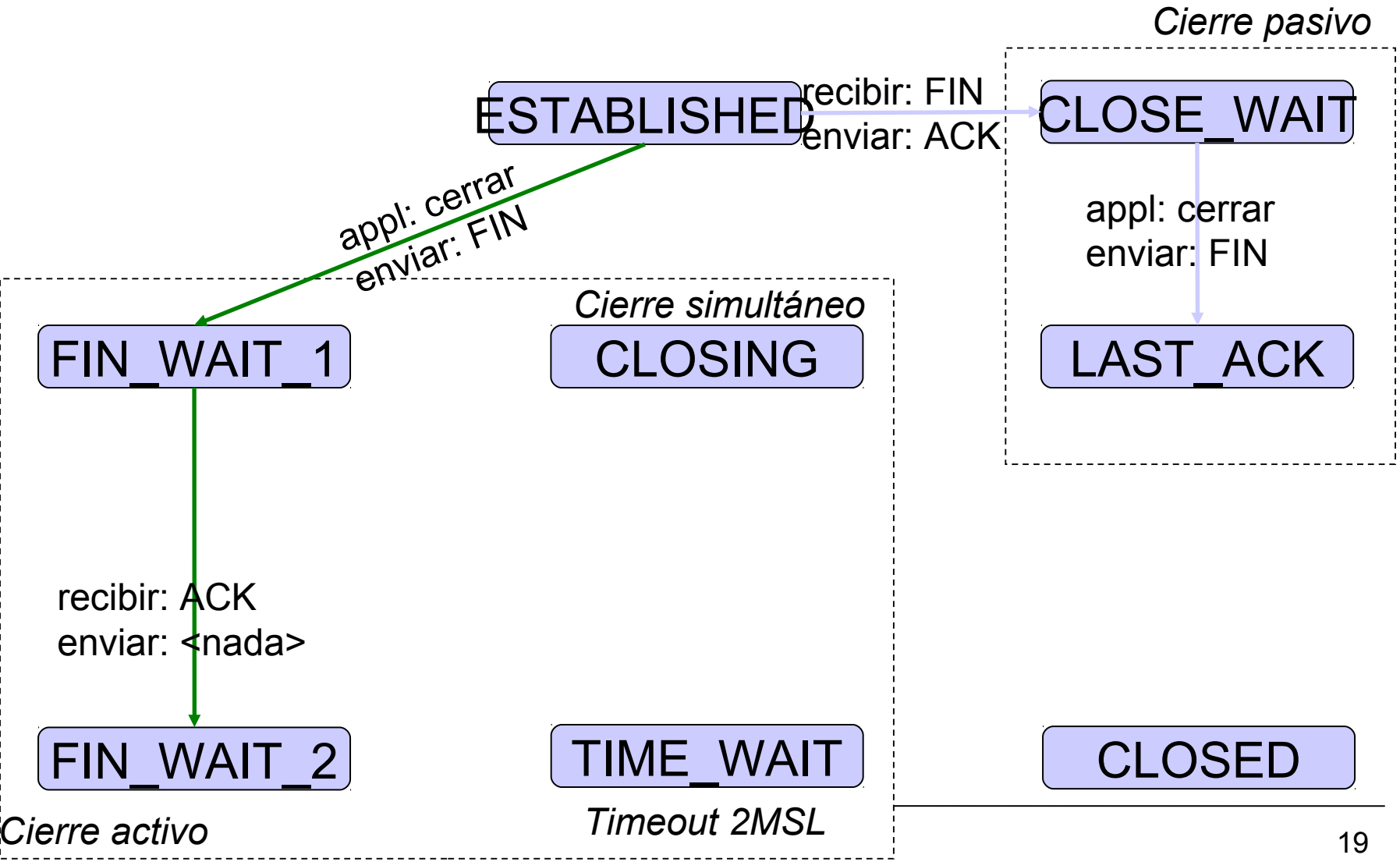
TCP: Diagrama de estados



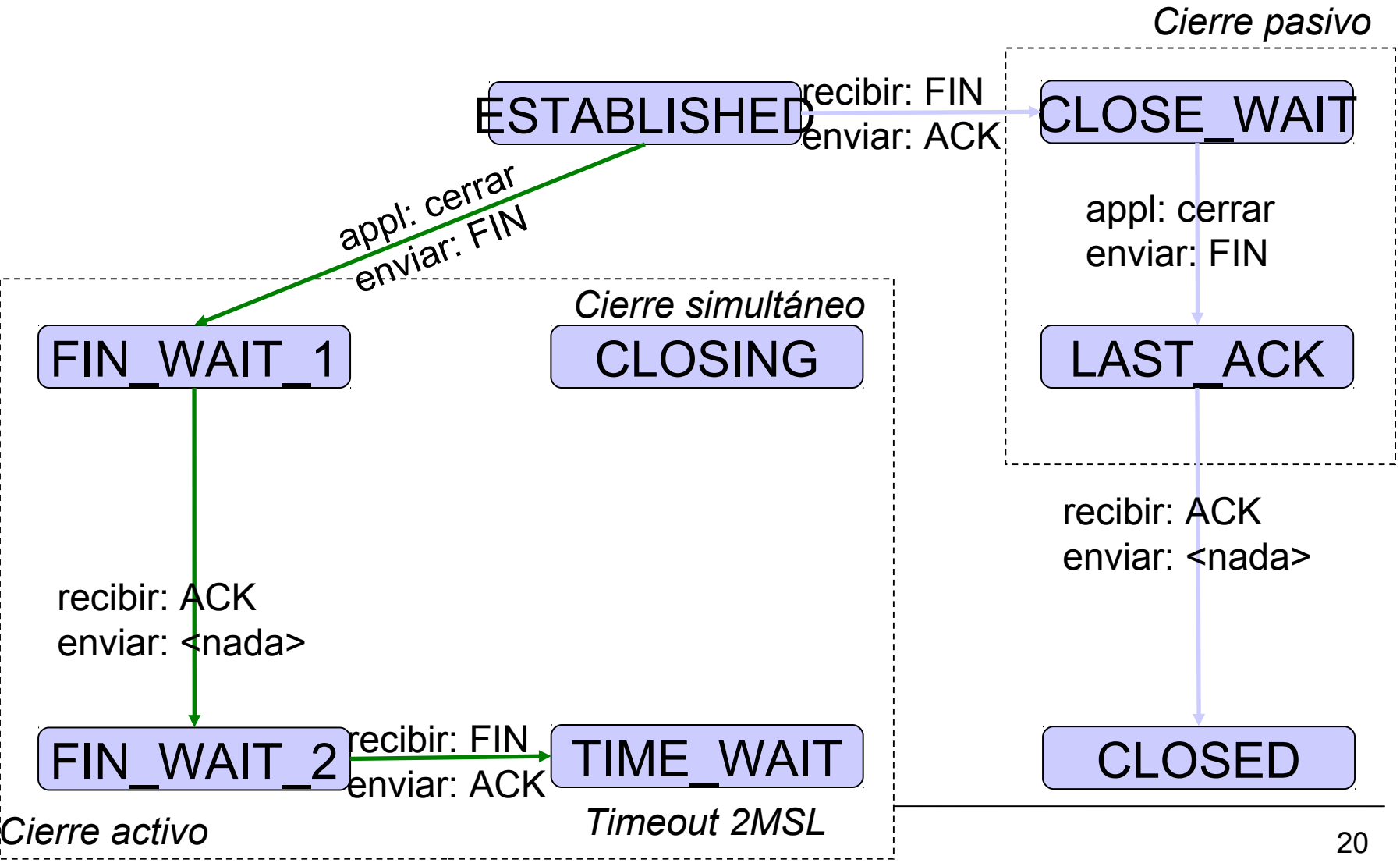
TCP: Diagrama de estados



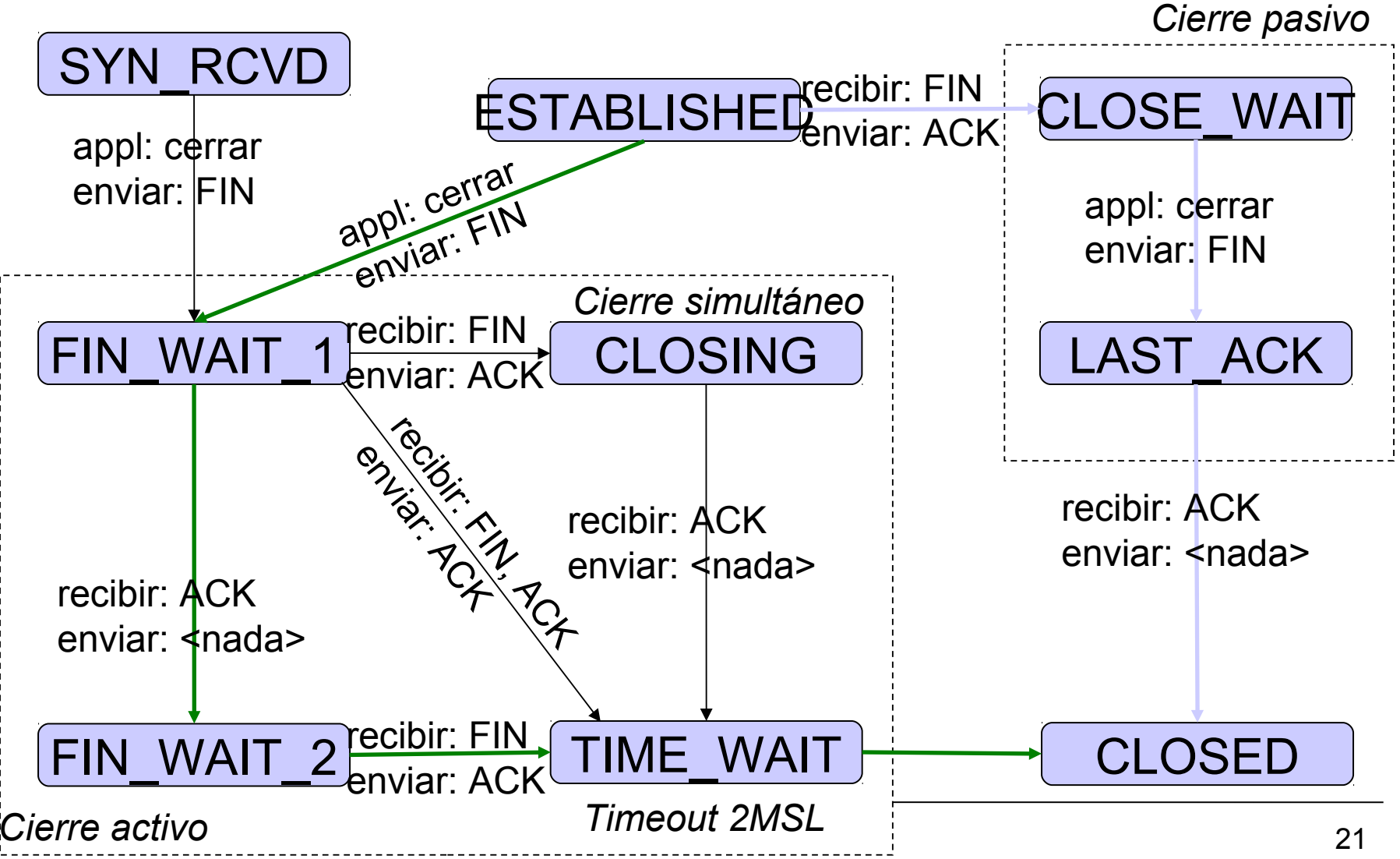
TCP: Diagrama de estados



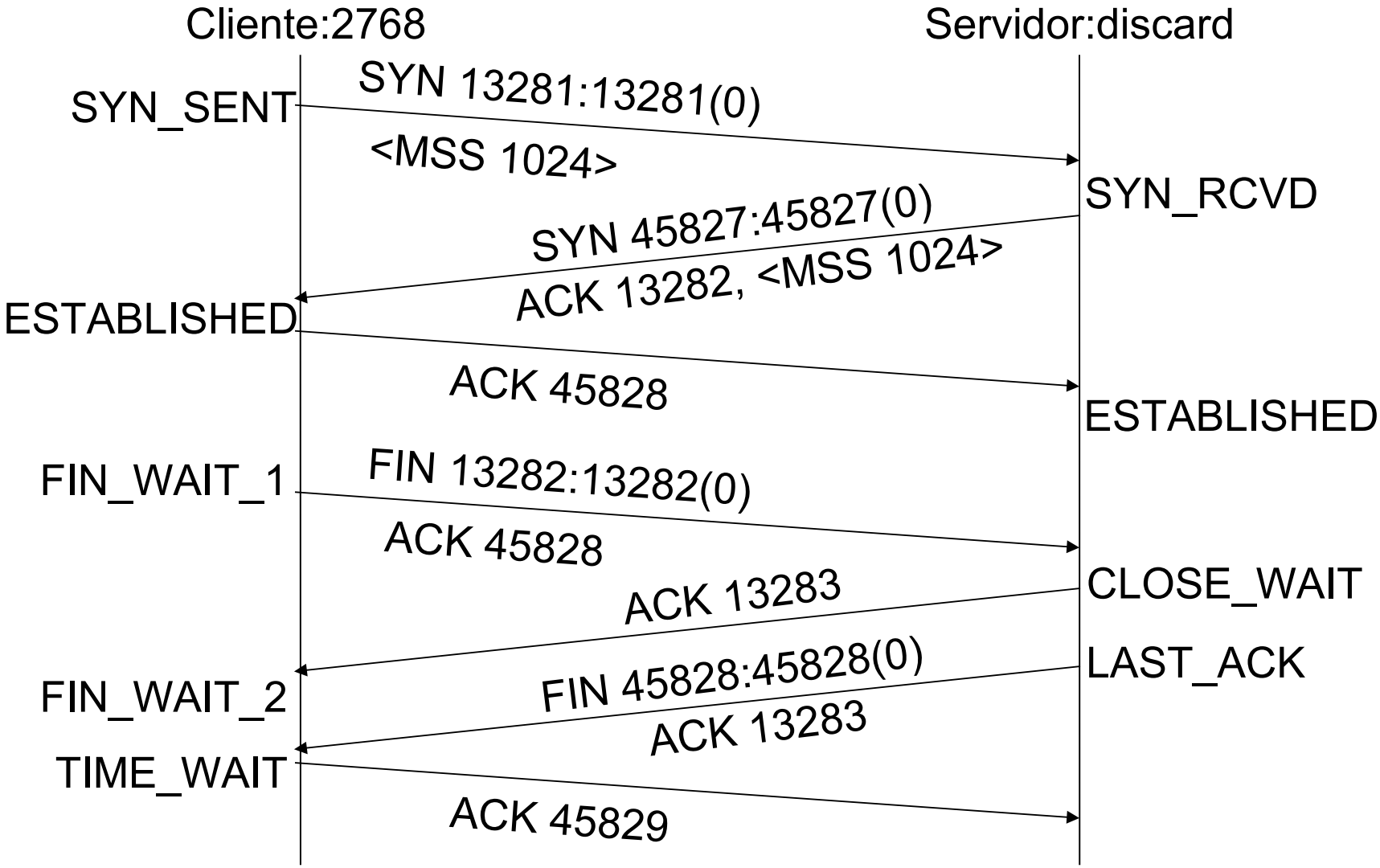
TCP: Diagrama de estados



TCP: Diagrama de estados



TCP: Diagrama de estados



TCP: Diagrama de estados

- Estado **TIME_WAIT**:
  - TCP espera 2 veces el tiempo máximo de vida de un paquete en la red (Maximum Segment Lifetime – MSL), por si se ha perdido el último ACK.
  - Variable /proc/sys/net/ipv4/tcp_fin_timeout (segundos)
  - Permite a TCP reenviar el ACK en caso de que se haya perdido (el otro extremo reenviará el FIN).
  - Mientras la conexión está en este estado, no se pueden reutilizar el par de sockets de esa conexión → Cualquier segmento retrasado recibido es descartado → Garantiza que no aparecen reencarnaciones de segmentos en futuras conexiones.

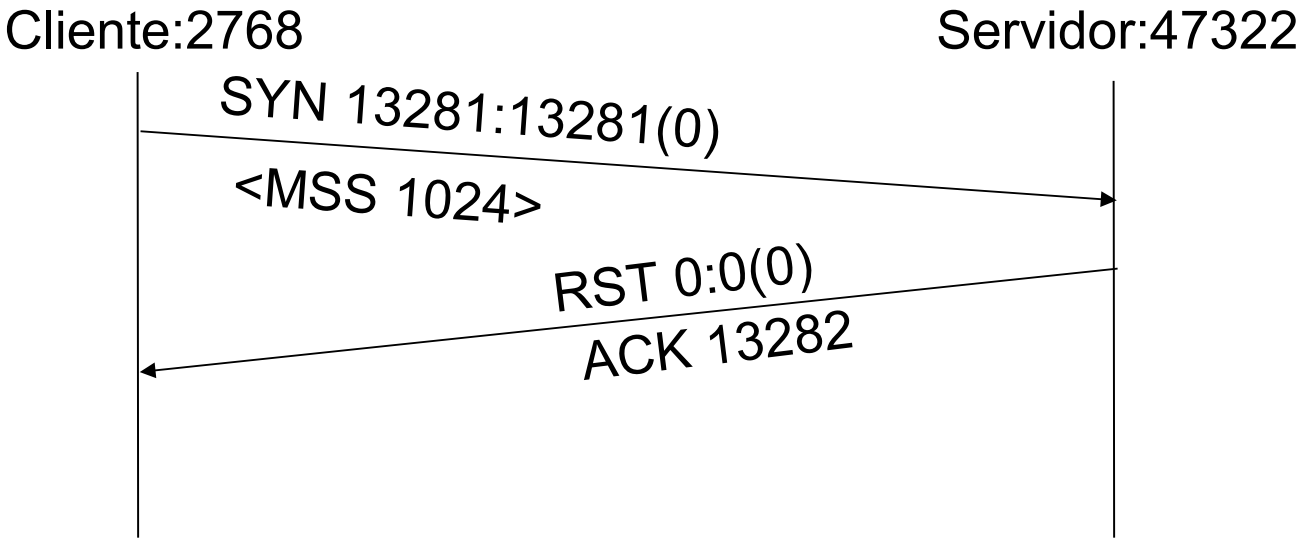
- Estado **FIN_WAIT_2**:
  - Permanecerá en este estado hasta recibir el FIN del otro extremo.
  - El otro extremo está en el estado CLOSE_WAIT y debe esperar a que se cierre la aplicación.
  - Para evitar una espera infinita las implementaciones establecen un tiempo de espera (misma variable que antes), tras el cual pasa directamente al estado CLOSED.

TCP: Segmentos de Reset

- Un segmento es de Reset cuando se activa en la cabecera TCP el flag RST.
- Se activa el bit de Reset en una conexión TCP cuando el paquete que ha llegado no parece, en principio, estar relacionado con la conexión a la que está referido el paquete.
- Las causas de generar un paquete con este bit para una conexión TCP pueden ser varias:
  - Intento de conexión a un puerto no existente
  - Abortar una conexión
  - Respuesta ante conexiones semi-abiertas

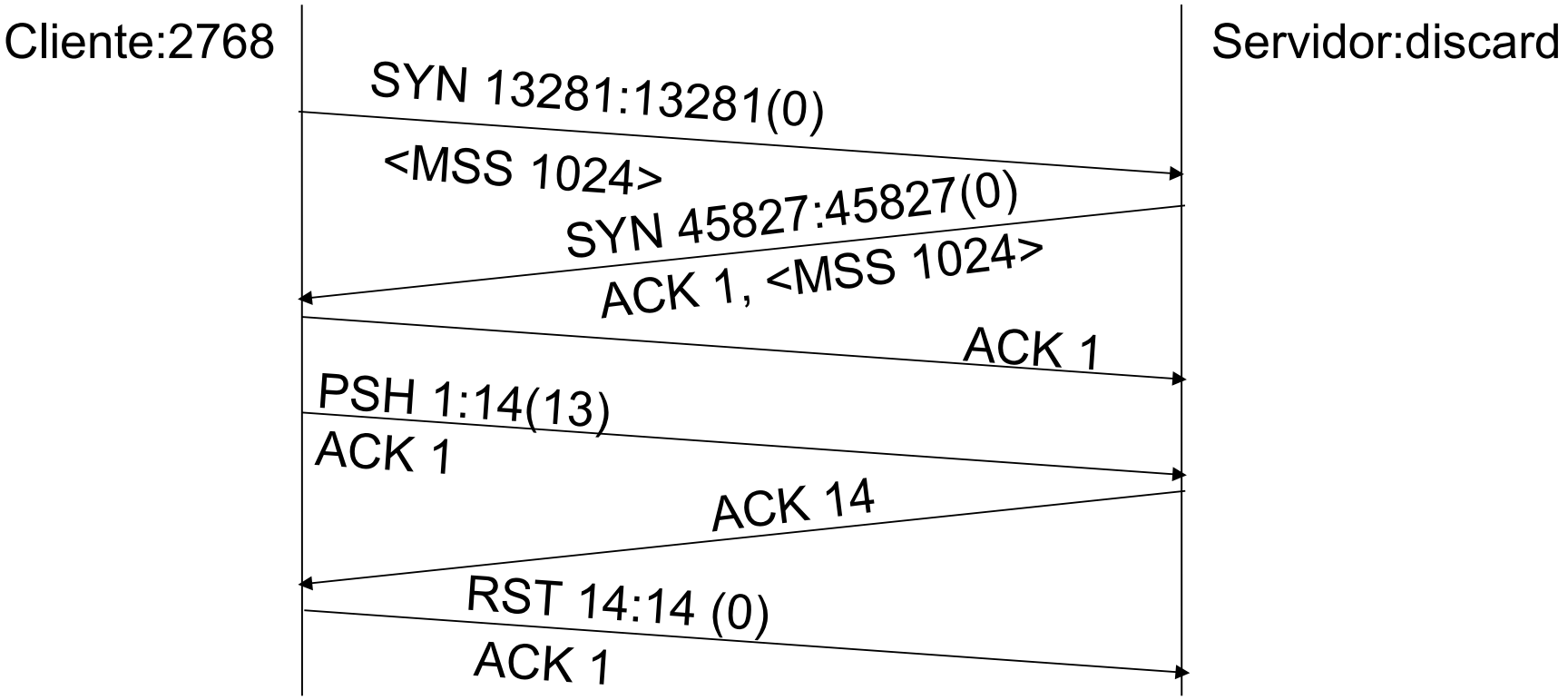
TCP: Segmentos de Reset

- Intento de conexión a un puerto no existente:
  - Si se trata de realizar una conexión a un puerto sin aplicación asociada → No hay ningún proceso escuchando una posible llegada de conexiones.



TCP: Segmentos de Reset

- Abortar conexión:
  - Existe la posibilidad de acabar mediante un paquete con el bit de RST (terminación anormal) → Cualquier dato esperando ser enviado será descartado automáticamente.



TCP: Segmentos de Reset

- Respuesta ante conexiones semi-abiertas:
  - Sucede cuando el servidor se cae y se vuelve a levantar, después de lo cual recibe datos del cliente.
  - La respuesta en este caso es un segmento de RESET